Visualization of process data with dynamic Bayesian networks

Tom Heskes Onno Zoeter
SNN, University of Nijmegen
Geert Grooteplein 21, 6252 EZ, Nijmegen, The Netherlands
{tom,orzoeter}@snn.kun.nl

Abstract

We describe a novel visualization algorithm for high-dimensional time-series data. The underlying model is a switching linear dynamical system, a particular variant of a dynamic Bayesian network. An important difference with most existing visualization techniques is the possibility to incorporate time dependencies between data points. Exact inference in switching linear dynamical systems is intractable, but new techniques for approximate inference enable fast computation of accurate posteriors. The model can be learned with a standard EM-algorithm. We illustrate our method on a real-world data set with sensor readings from a paper machine.

1 Introduction

Industrial processes are getting more and more complex and there is a growing need for autonomous systems that control them. For this we need robust online monitoring and diagnosis tools: we want to detect abnormal behavior and, if possible, diagnose the failure. Dynamic Bayesian networks are a relatively new tool, similar in spirit to the well-established Kalman filter, and seem perfectly suited for this task. In this article, we will focus on the use of dynamic Bayesian networks for the visualization of process data. See e.g. [16, 19, 15] for other applications in fault detection and diagnosis.

A dynamic Bayesian network models the complex hybrid system (e.g., the paper machine or production plant) as a probability distribution of states over time. It is a state-space model, similar to the well-known Kalman filter [13]. The Kalman filter is traditionally *the* tool in signal processing and control, and works fine for systems with linear dynamics and Gaussian noise. In reality, real-world systems have many nonlinearities and more advanced dynamic Bayesian networks are required to accurately model them.

The so-called hybrid dynamic Bayesian networks, which consist of both continuous and discrete ("switch") nodes, are particularly useful for representing industrial process data. The discrete switch nodes accommodate different regimes or modes of operation and the continuous nodes describe the (continuous) system dynamics. The switching linear dynamical systems [18, 9] a.k.a. switching Kalman filters form a particular subclass of the hybrid dynamic Bayesian networks. They basically correspond to a different Kalman filter for each different mode. Switching linear dynamical systems have many applications in many different fields, ranging from the modeling of financial time series (there also called Markov switching models [14]) to tempo tracking and rhythm detection in music [6].

2 Switching linear dynamical systems

2.1 Description

Figure 1 shows a graphical representation of a switching linear dynamical system. As in the original Kalman filter, the latent or hidden variables $\mathbf{x}_t$ are thought to form a (lower-dimensional) representation of the *state* of the system. The dynamics of these latent variables is a kind of Brownian motion. The typically high-dimensional observations ("sensor readings") $\mathbf{y}_t$ are modeled as a direct (linear) function of these latent variables with additional Gaussian noise. The dynamic switches s_t represent the mode or *regime* of the system at (discretized) time t. The state of this switch determines both the dynamics between the latent variables $\mathbf{x}_t$ and the relationship between these latent variables and the observations $\mathbf{y}_t$. The switches follow a first-order Markov process.

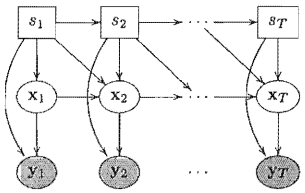

Figure 1: A switching linear dynamical system. The dynamic switches s_t determine both the dynamics in the continuous latent variables $\mathbf{x}_t$ and the link between $\mathbf{x}_t$ and the observations $\mathbf{y}_t$.

2.2 Visualization

In this article, we focus on the use of switching linear dynamical systems for visualization purposes. Our model is a dynamic generalization of the (static) mixture of probabilistic principal component analyzers [4]. Probabilistic principal component analysis is functionally equivalent to standard principal component analysis. With static switches we obtain a mixture model (Figure 1, but then without the links between the time slices), and with dynamics between switches and continuous latent variables we have a switching linear dynamical system. For visualization purposes, the dimension of the latent variables $\mathbf{x}_t$, that can be loosely interpreted as projections onto the principal components, is taken to be two-dimensional. Each switch state corresponds to a different subplot: the more different states a switch can have, the more subplots. Furthermore, the noise on the observations $\mathbf{y}_t$ is taken to be spherical (independent, with the same variance in all dimensions). In mathematics, the switching linear dynamical systems that we consider in this article have the following specifications:

$$
\begin{aligned}
p(\mathbf{x}_{t+1}|\mathbf{x}_t, s_t = i, s_{t+1} = j, \boldsymbol{\theta}) &= \begin{cases} \mathcal{N}(\mathbf{x}_{t+1}; A_j\mathbf{x}_t, Q_j) : i = j \\ \mathcal{N}(\mathbf{x}_{t+1}; \mathbf{0}, \Sigma_j) : i \neq j \end{cases} \\
p(\mathbf{y}_t|\mathbf{x}_t, s_t = i, \boldsymbol{\theta}) &= \mathcal{N}(\mathbf{y}_t; C_i\mathbf{x}_t + \boldsymbol{\mu}_i, r_i I) \\
p(s_{t+1} = j|s_t = i, \boldsymbol{\theta}) &= \Pi_{i\rightarrow j} \, .
\end{aligned}
$$

Here $\boldsymbol{\theta} = \{\Pi_i, A_i, Q_i, \Sigma_i, C_i, \boldsymbol{\mu}_i, r_i | i = 1 \ldots M\}$, with M the number of regimes, refers to the complete set of model parameters and $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma)$ stands for a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ. It can be seen that $\mathbf{x}_t$, the state of the system, is "reset" after a regime switch, effectively decoupling the principal components of the different regimes.

The basic idea of visualization is now as follows. Given a set of model parameters $\boldsymbol{\theta}$ and observations $\mathbf{y}_{1:T}$, we compute for every time t both the posterior probability $p(s_t = i|\mathbf{y}_{1:T}, \boldsymbol{\theta})$ of being in regime i and the posterior means $E[\mathbf{x}_t|s_t = i, \mathbf{y}_{1:T}, \boldsymbol{\theta}]$ of the continuous latent variables conditionalized on the regime state. In principle, each observation $\mathbf{y}_t$ is projected onto each subplot i at the posterior mean $E[\mathbf{x}_t|s_t =$

$i, \mathbf{y}_{1:T}, \boldsymbol{\theta}]$. The regime posterior $p(s_t|\mathbf{y}_{1:T}, \boldsymbol{\theta})$ specifies the "amount of ink" that is used. Alternatively, one can choose to plot only those projections that have regime posteriors above a preset threshold.

In Figure 2, each level corresponds to a different switching linear dynamical system. The top level in fact corresponds to a "standard" Kalman filter: there is only one switch state. The second level has 4 subplots and thus 4 different regimes. The model corresponding to the lowest level is slightly more complex, and can be thought of as 4 independent switching linear dynamical systems with respectively 2, 3, 2, and 2 regimes. This hierarchy of subplots allows the user to recursively zoom in on (apparently) interesting behavior. Construction the third and subsequent levels in the same order complexity as the second level requires some additional and natural constraints, that we will not present in detail here.

3 Inference and learning

3.1 Approximate inference

Exact inference in switching linear dynamical systems is intractable: it scales exponentially with the number of time slices. That is, in general the exact posterior is a mixture of M^T Gaussians, with M the number of regimes. Approximate inference algorithms can be divided into two classes: stochastic sampling approaches, like (Rao-Blackwellized) particle filtering [8] and Markov chain Monte Carlo [5], and deterministic variational approaches [9]. Of these variational approaches, expectation propagation [17] seems particularly suited for our purposes. The filtering pass of this approximation is known as generalized pseudo Bayes 2 (GPB2) [1]. The basic idea is to approximate $p(\mathbf{x}_t|s_t = i, \mathbf{y}_{1:T}, \boldsymbol{\theta})$ with a single Gaussian instead of keeping the exact mixture. In the context of visualization, where we are only interested in the posterior mean, this is well justified.

The intuition behind the method is best understood by considering the filtering pass. The exact posterior $p(\mathbf{x}_1|s_1 = i, \mathbf{y}_1, \boldsymbol{\theta})$ is a single Gaussian, but the posterior $p(\mathbf{x}_2|s_2 = i, \mathbf{y}_{1:2}, \boldsymbol{\theta})$ is a mixture of two Gaussians (one for the case $s_1 = i$ and one for $s_1 \neq i$). Instead of propagating this mixture to the third time slice (and thus introducing a rapid growth of complexity in time) we approximate it by a single Gaussian closest in Kullback-Leibler divergence to the original mixture. This "collapse operation" boils down to moment matching. A recursive filtering procedure based on such a collapse keeps M Gaussians, one for each switch state, in every time slice.

In [12] we derive a similar smoothing pass. It is shown that expectation propagation can be understood as a kind of approximate belief propagation. In belief propagation, beliefs are computed by propagating messages. The important notion in expectation propagation is that the additional collapse operation should work not on the messages themselves, but on the beliefs. Message updates can then be derived from the approximate beliefs (see [12] for details).

3.2 Learning

Given a data set $\mathbf{y}_{1:T}$, we can find the maximum likelihood settings of the parameters in the switching linear dynamical system

$$\boldsymbol{\theta}_{\mathrm{ML}} = \operatorname*{argmax}_{\boldsymbol{\theta}} p(\mathbf{y}_{1:T}|\boldsymbol{\theta}) \,,$$

using an EM-algorithm [7]. In the Expectation step, we have to compute posteriors like the ones required for the visualization itself given the current setting of the parameters $\boldsymbol{\theta}^{\mathrm{old}}$, i.e., $p(\mathbf{x}_t|s_t = i, \mathbf{y}_{1:T}, \boldsymbol{\theta}^{\mathrm{old}})$ and $p(s_t = i|\mathbf{y}_{1:T}, \boldsymbol{\theta}^{\mathrm{old}})$, but also

two-slice beliefs such as $p(\mathbf{x}_t, \mathbf{x}_{t+1}|s_t = i, s_{t+1} = i, \mathbf{y}_{1:T}, \boldsymbol{\theta}^{\text{old}})$ and $p(s_t = i, s_{t+1} = j|\mathbf{y}_{1:T}, \boldsymbol{\theta}^{\text{old}})$. As noted above, exact computation is infeasible and expectation propagation can be used to compute approximate beliefs. Given these beliefs the Maximization step, maximizing the complete data loglikelihood

$$\hat{\mathcal{L}}(\mathbf{y}_{1:T}|\boldsymbol{\theta}) = E_{p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T},\boldsymbol{\theta}^{\text{old}})}\left[\log p(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}|\boldsymbol{\theta})\right] ,$$

w.r.t. $\boldsymbol{\theta}$, is relatively straightforward (see e.g. [10] for a description of the EM-algorithm for linear dynamical systems without switches). The EM-algorithm is guaranteed to converge to a local minimum of the likelihood and can easily account for missing observations.

4 Paper mill data

We present an example from a board production process. Board is made as a continuous web at production speeds ranging from 600 to 800 meters per minute. Multiple paper types are produced on the same line. The changes of type are performed without stopping production. The process tends to be complex, differing from machine to machine and, in general, is only partially understood. In this domain a visualization method may be useful both as a tool for novelty detection and as an on-line monitoring tool.

In Figure 2 ten hours of production data is presented. The data consists of 13 crucial sensor readings such as machine speed, steam pressures, flows of additives etc. measured once a minute. The colors or gray scales in the figure indicate different types of paper (added after learning, i.e., this information is not used in any of the algorithms).

We see that macro scale clusters such as the paper type are well separated. Static models are able to capture these differences as well. The benefit of the dynamic model is more apparent in the lower levels where drifts in the state of the process can be observed. The bars below the subplots show that there is a clear tendency to zoom in on data that is clustered in time (see for example the subdivision of the first parent plot at the second level into the children subplots at the third level).

5 Conclusion and discussion

In our opinion, (hybrid) dynamic Bayesian networks are a fruitful tool for modeling and monitoring complex industrial processes. The probabilistic framework is ideally suited to combine both prior knowledge and data. In dynamic Bayesian networks, the problem of diagnosis and monitoring has been reduced to the problem of tracking, or, more generally, of computing posterior probabilities. This is not an easy problem: exact probabilistic inference is practically infeasible in most nonlinear systems. The rapid development of algorithms for approximate inference, both sampling (particle filters) and variational approaches (expectation propagation), will help to increase the popularity of dynamic Bayesian networks and their practical applications in the near future.

The visualization tool that we described in this article is "just" an example. It extends previous work [4] on the visualization of static data to the dynamic domain. On static data, our algorithm will roughly give the same visualization as the static mixture of principal component analyzers. On dynamic data, our algorithm uses neighboring observations to obtain a better tracking of the state of the system. Compared with the static approach, our algorithm tends to construct "swarms" rather than "blobs" and more clearly separates different types of dynamic behavior.

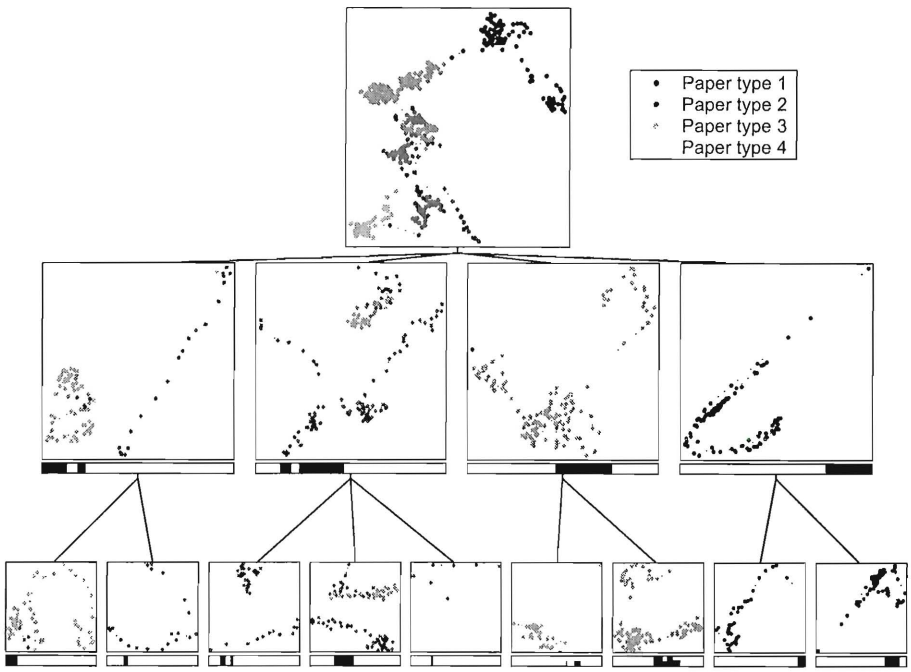

Figure 2: Ten hours of production data from a paper mill projected using a hierarchy of switching linear dynamical systems. The labels encode different paper types. Bars below the subplots visualize the probabilities $p(s_t|\mathbf{y}_{1:T})$ as a function of time. Note that the labels themselves are not used by the algorithm.

Use of the approximate inference algorithm described in Section 3.1 makes learning and inference of about the same order complexity as the static variant.

Recently developed alternatives are the GTM through time [3] and similar algorithms based on self-organizing maps. In these models, the continuous latent variables are considered stationary, restricting the dynamics to transition probabilities between the nodes that they belong to. Furthermore, the structure of these models has to be determined in advance and does not give the user flexibility to interactively zoom in on the data.

Obviously, there are many ways to improve upon the current tool. Currently, it is interactive. The user is asked to click on (apparent) clusters in a parent plot. The number of clicks determines the number of children subplots and the locations are translated to an appropriate parameter initialization in the EM-algorithm for learning the parameters of these subplots. Ideally, this would all be automatic: the algorithm itself should find out the optimal number of clusters and the corresponding (initial) cluster centers. On a more general level, the restriction to a two-dimensional state space, here required for visualization, may be inappropriate. Also here one would perhaps an automatic tool for computing the optimal number of principal components. In principle, this can all be done in a Bayesian setting, again either with sampling approaches, like reversible jump MCMC [11], or with variational procedures similar to those in e.g. [2].

Acknowledgments

We acknowledge support from the Dutch Competence Centre Paper and Board and thank Ali Taylan Cemgil and Alexander Ypma for many helpful discussions.

References

[1] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation and Tracking: Principles, Techniques, and Software.* Artech House, 1993.

[2] Matthew Beal and Zoubin Ghahramani. Propagation algorithms for variational Bayesian learning. In *Advances in Neural Information Processing Systems 13*, 2001.

[3] C. Bishop, G. Hinton, and I. Strachan. GTM through time. In *IEE International Conference on Artificial Neural Networks*, pages 111–116, Cambridge, 1997.

[4] C. M. Bishop and M. E. Tipping. A hierarchical latent variable model for data visualization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3), 1998.

[5] C. Carter and R. Kohn. Markov chain Monte Carlo in conditionally Gaussian state space models. *Biometrika*, 83:589–601, 1996.

[6] A. Cemgil, B. Kappen, P. Desain, and H. Honing. On tempo tracking: Tempogram representation and Kalman filtering. In *Proceedings of 2000 International Computer Music Conference*, pages 352–355, Berlin, 2000. International Computer Music Association.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[8] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[9] Z. Ghahramani and G. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12:963–996, 1998.

[10] Zoubin Ghahramani and Geoff Hinton. Parameter estimation for linear dynamical systems. Technical report, University of Toronto, 1996.

[11] P. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[12] T. Heskes and O. Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In *Proceedings UAI-2002*, 2002.

[13] R. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME (Journal of Basic Engineering)*, 82D:35–43, 1960.

[14] C. Kim. Dynamic linear models with Markov-switching. *Journal of Econometrics*, 60:1–22, 1994.

[15] U. Lerner, B. Moses, M. Scott, S. McIlraith, and D. Koller. Monitoring a complex physical system using a hybrid dynamic Bayes net. In *Proceedings UAI-2002*, 2002.

[16] U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *AAAI/IAAI*, pages 531–537, 2000.

[17] T. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2001)*. Morgan Kaufmann Publishers, 2001.

[18] K. Murphy. Learning switching Kalman-filter models. Technical report, Compaq CRL., 1998.

[19] R. Sterritt, A. Marshall, C. Shapcott, and S. Mcclean. Exploring dynamic Bayesian belief networks for intelligent fault management systems. In *Proc. IEEE Int. Conf. Systems, Man And Cybernetics*, volume V, pages 3646–3652, 2000.